

Graphical Programming, Simulation and Control System for Dual-Arm Mobile Robot

XU Hua¹, CAO Qi-xin¹, QIU Chang-wu¹, Ikuo Nagamatsu², Kazuhiko Yokoyama²

(1. Research Institute of Robotics, Shanghai Jiao Tong University, Shanghai 200052, China;

2. Corporate R&D Center, YASKAWA Electric Corporation, Kitakyushu-City 803-8530 Japan)



Abstract: A graphical programming, simulation and control system is described which can provide Dual-Arm robot a task planning, 3D simulation and control environment. User can program a Dual-arm robot through graphical programming, preview and check motion simulation in 3D simulation, and then download optimal program to actual robot controller to ensure the desired results. The system uses Java3D and VRML for 3D simulation, and runs on RT-Linux platform to ensure Real-Time in simulation and control. Multi-robot mechanism is realized to support pipelining work or cooperation between robots. This system now supports Dual-Arm Mobile robot of YASWAKA Electric Corporation.

Key words: graphical programming; simulation and control; dual-arm mobile robot; Java3D; RT-Linux; virtual reality modeling language (VRML)

双臂移动机器人的图形化编程与仿真控制系统

徐 化¹, 曹其新¹, 邱长伍¹, 长孙郁男², 横山和彦²

(1. 上海交通大学机器人研究所, 上海 200052;

2. 日本安川电机株式会社 机器人技术开发部, 北九州 803-8530, 日本)

摘 要: 介绍一种双臂移动机器人的图形化编程与仿真控制系统。用户通过图形编程界面对机器人编程和任务规划, 在三维仿真环境中预览和分析规划结果, 最后将优化程序下载到真实机器人中进行控制。采用 Java3D 和 VRML 实现三维仿真; 基于 RT-Linux 平台保证仿真控制的实时性; 实现了多机器人机制以支持机器人的流水线协作。该系统现在支持日本安川电机株式会社的双臂移动机器人产品。

关键词: 图形化编程; 仿真控制; 双臂移动机器人; Java3D; RT-Linux; 虚拟现实建模语言(VRML)

中图分类号: TP391.9

文献标识码: A

文章编号: 1004-731X (2006) 09-2541-05

Introduction

Robot is in an increasing use into fields of replacing human to fulfill complex and dangerous tasks. Dual-Arm mobile robot, as an especial robot whose two arms can execute most of human tasks while reserving high mobility to the desired destination, meets the future needs of high dexterousness and reliability. However, task planning and control of Dual-Arm mobile robot is uneasy to execute. It commonly suffers from a large amount of input parameters and cooperation difficulties in control. An easy way has to be found to program and

control Dual-Arm mobile robot. At the same time, robot research work can be shifted away from production to avoid interruptions and downtimes in robot plant.

Graphical Simulation provides robot a virtual environment for offline programming and simulation. In 2000 a Web based robot Simulation System using VRML was proposed by Martin Rohrmeier^[1]. Many similar systems are for educational purposes, as Robot Draw by Melinda F^[2] et al, KUKA^[3] robot simulation system by Vollmann K. Some are designed for mathematical simulation, as SRTK for Puma 560 robot proposed by Dixon WE^[4] et al. These traditional softwares are mainly designed for industrial manipulator simulation, they don't contact controller closely. Simulation and task planning for complex motion of Dual-Arm mobile robot are not achieved. All these bring to Dual-Arm mobile robot users potential inconvenience and fallibility.

收稿日期: 2005-07-04

修回日期: 2006-01-02

基金项目: 国家自然科学基金资助项目 (60443007)

作者简介: 徐化(1978-), 男, 上海人, 硕士生, 研究方向为机器人学, 机电一体化技术, 智能控制; 曹其新(1960-), 男, 上海人, 博士, 教授, 博导, 研究方向为机器视觉与识别, 智能控制, 多智能体协调; 邱长伍(1978-), 男, 甘肃人, 博士生, 研究方向为机器人学, 机电一体化技术, 系统仿真。

Aimed above inconvenience, this paper presents a graphical programming, simulation and control system. (1)The system runs on PC system. It provides an easy way of Dual-Arm mobile robot programming and simulation, its controller part serves as GUI for actual robot controller. (2)Developed on RT-Linux, the system realizes real-time simulation. (3) Based on Java, the system realizes "Write Once, Run Anywhere." with low maintenance costs. (4) Multi-robot mechanism is realized to simulate or control several robots at the same time, which is crucial important in complex cooperation such as pipelining work. These advantages endow Dual-Arm robot with a strong meaning of application. The system was tested on YASKAWA Electric Company's product SmartPal which is a Dual-Arm mobile robot. The system will support more robot products in the future.

1 System Architecture

This paper realized a new architecture to divide the system into two sub-ones: Programming and Simulation; robot controller. In Programming and Simulation part, user make offline programming, and then analyze motion in simulation to preview program results. In controller part, user applies optimal program from analysis into actual control. Thus, system attained high efficiency in robot R&D process as well as increasing productivity in industrial applications.

System architecture is based on 4 Layers: User Interface, Java3D and Linux, RT-Linux Kernel and hardware (see Fig. 1).

1.1 User Interface

System provides a friendly interface where user can initialize environment such as numbers and positions of robot. Then, user makes task planning and offline program in programming panel as an easy graphical programming environment. With the program user can further control the real robot. In Viewing Platform, user observes robot motion in simulation, analyze and predict potential problems.

1.2 Java3D and Linux Layer

The system was developed in Java on Linux platform. Java is used to develop human-machine interaction part; Java3D is used to develop 3D simulation in robot modeling, motion animation and environment construction, as well as rendering,

lighting and action-event response.

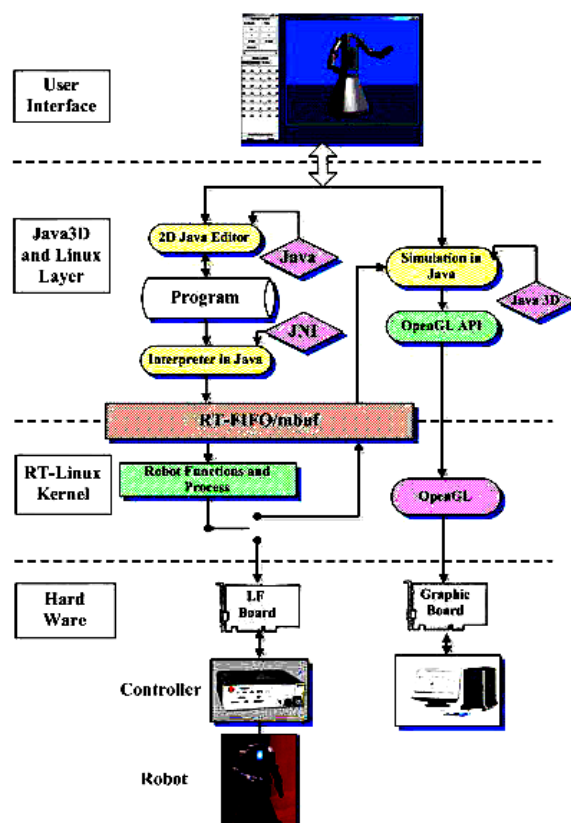


Fig. 1 System Architecture

1.3 RT-Linux Kernel Layer

Real-Time must be realized to ensure strict coherence between simulation and actual robot motion. JNI (Java Native Interface) was applied to ensure transfer between system and actual robot controller, driving file of controller is encapsulated in Robot API by C language that reads and writes controller physically. Then, system task is divided into Real-Time task and non Real-Time task in RT-Linux. RT_FIFO (first in, first out) is used to communicate between Real-Time task and non Real-Time task to ensure that Real-Time task will be never blocked, thus ensure accurate simulation.

1.4 Hard Ware Layer

Robot API accesses the robot controller directly. It is tie between system and actual robot. However, it's very fussy for user to program a Dual-Arm robot via machine language. Luckily, JNI provides us the communication function between Java and native resources. Using JNI to integrate Robot API into system, system realize convenient graphical programming environment for user to control the actual robot.

2 Robot Simulation

2.1 Architecture of Virtual Environment

The architecture of virtual environment was realized which is supported by complex shapes, color effect, light effect, transform effect etc. In Java3D, scene graph is described in dendriform structure where all kinds of node (shape node, light node etc.) are seen as leafs while relations between these nodes are seen as branches. Scene graphs will be finally shown in a common universe. In the system, initial root of virtual environment is Locale following by 3D graph root node (See Fig. 2). A light source node was attached on the graph root node that is always invariable. A Viewing Platform node attaches graph root node via one Transform graph node and one Branchgraph node to support viewing navigation. When Transform graph node's pose changes, Viewing Platform's pose follows the change.

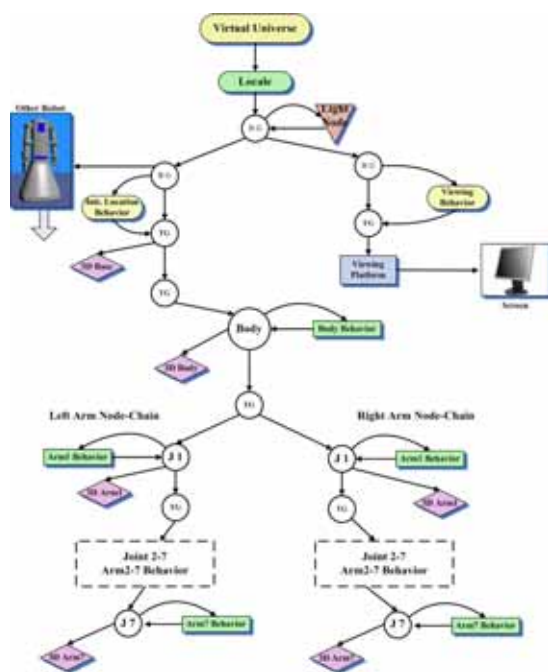


Fig. 2 Architecture of Virtual Environment

2.2 Robot Construction in Virtual Environment

A Dual-Arm mobile robot in the system is seen as 19 connecting rods structure having 21 degrees-of-freedom (DOF), neighboring rods are connected with one joint. Virtual robot is constructed in java3D. Because common modeling in Java3D is very trivial and the shape will be distorted, virtual robot was decomposed into several basic parts which were described by VRML (Virtual Reality Modeling Language). Unigraphics NX was used to build the complex shape of robot, then the shapes of robot was

transformed to VRML2.0 files. Java3D load VRML files into virtual environment through VrmlLoader to finish modeling.

To assemble 21 DOF robot in virtual environment, basic robot parts were loaded to right position in Cartesian coordinate in virtual environment to ensure every two basic parts assemble at corresponding Joint. In Java3D, these basic parts are defined as BranchGroup while robot Joints defined as TransformGroup. One coordinate was attached on each Joint node. Robot assemblage in virtual environment was realized in this way which provided reference for motion simulation.

Shown in Fig. 3, Joints and parts of Dual-Arm mobile robot are described as Node chain in Java3D.

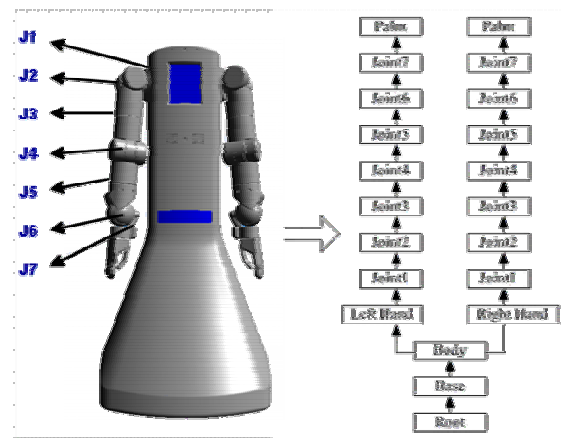


Fig. 3 Architecture of Robot basic Nodes

2.3 Robot Kinematics and its Implementation

In simulation, virtual robot's motion obey right kinematics rule. In describing translation and rotation relations between neighboring robot rods, D-H parameters [8] was used to calculate transformation between rods. One coordinate frame was setup on each rod to form a coordinate frame chain, and then use Homogeneous transformation to describe relative position and gestures of these coordinate frames.

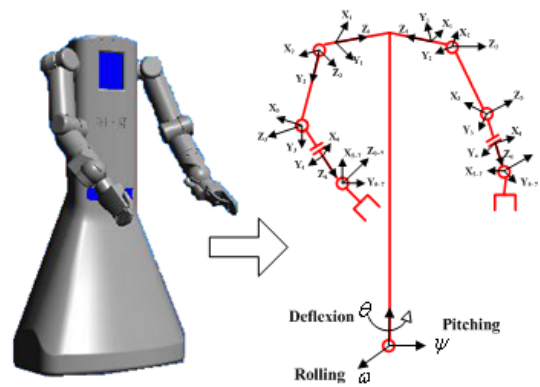


Fig. 4 Coordinate Chain on Robot Rods

In D-H parameters, relation between Rodi-1 and Rodi can be described with four Homogeneous transformations. Knowing these Homogeneous transformations, the transformation Matrix is easily calculated

$$A_i = Rot(z, \theta_i) Trans(0, 0, d) Trans(a_i, 0, 0) Rot(x, \omega_i)$$

$$= \begin{bmatrix} C\theta_i & -S\theta_i C\omega_i & S\theta_i S\omega_i & a_i C\theta_i \\ S\theta_i & C\theta_i C\omega_i & -C\theta_i S\omega_i & a_i S\theta_i \\ 0 & S\omega_i & C\omega_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

The transformation from coordinate_n to coordinate_{i-1} can be calculated

$${}^{i-1}T_n = A_i A_{i+1} \cdots A_n \quad (2)$$

In this way system realized robot moving and motion simulation.

2.4 Multi-Robot

Multi-threads in Java is used to realize multi-robot. Two threads were set up as Manipulate thread and Detect thread. 19 joint angles and distance in x and y directions are the parameters to be considered in robot motion detecting. So a robot parameter class was created as a parameter pool to store data of joint angles and distance in either directions in motion simulation. Manipulate thread and detect thread works alternately to access parameter pool. They are producer and consumer in relationship (see Fig. 5).

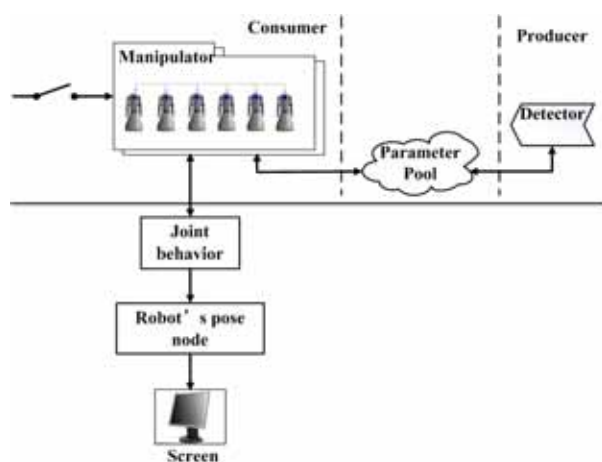


Fig. 5 Multi-Thread

Detect thread calls Robot API and gets the current robot information, then update data in parameter pool. Manipulate thread check the parameter pool constantly, as soon as data change Manipulate thread reads data and then awake robot's behavior. When multi robots constructed, the corresponding detect thread and manipulate thread

were constructed, so the function can be normally implemented to fields using multi-robot.

2.5 Application

In this application, real robot SmartPal will hand over a piece of paper to the guest. At the same time, virtual robot will work in the simulation system in synchronization with real robot. Fig. 6 shows the application example. These pictures were taken from World robot Exhibition held in Kita-Kyushu, Japan.



Fig. 6 Virtual Robot Work with Real Robot in Synchronization

3 Robot Real-Time Control

3.1 Robot Control

For precise and stable control, driving files of robot controller are coded in C language, namely Robot Application Programming Interface (Robot API). Robot API can access the robot controller physically (such as read and write hardware), it's the tie between the system and actual robot. However, it's a very fussy for operator to program robot via this machine language. JNI is used to integrate Robot API in system to realize communication between Java and robot controller, forming a graphical environment for user to program and control the robot.

3.2 Using JNI in Robot Control

JNI is the native method interface between Java and native resources; it enables Java to call libraries or application programs coded in other languages. In the system, the API to robot controller is coded in C. JNI integrates Robot API to the system as Fig. 7 shows.

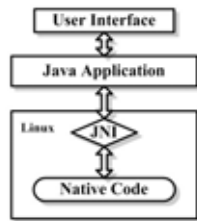


Fig. 7 Using JNI in system develop

3.3 Real-Time in RT-Linux

The system was developed on RT-Linux system to ensure Real-Time of Robot simulation and control. RT-Linux Real-Time process management mechanism is seen in Fig.8.

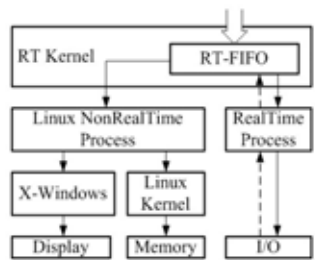


Fig. 8 RT-Linux Real-Time Process Manage Mechanism

System was divided into two of Real-Time and nonReal-Time parts. RT-Linux provides API to compile Real-Time part of the system into Kernel module and then load this module into RT-Linux Kernel which directly charges hardware interrupt. The nonReal-Time part of the system will be executed on User Side. Priority Attempter of RT-Linux was utilized to ensure Real-Time control. RT-Linux is embedded with RT-FIFO (first in, first out) queue, which is seen by Linux as Character Device, named /dev/rtf0, /dev/rtf1..., the maximal number of RT-FIFO is set in system Kernel Compilation. RT-FIFO charges the communication between Real-Time task and common Linux process, and it will never be maneuvered out when memory page changes, thus ensure no blocking in Real-Time task. When RT-FIFO is constructed, main program in Kernel is running in periodic Real-Time thread mode. It means that the system periodically executes data capture, export control result, and exchange data with nonReal-Time process in User Side. Fig. 9 shows Real-Time Control process in the system.

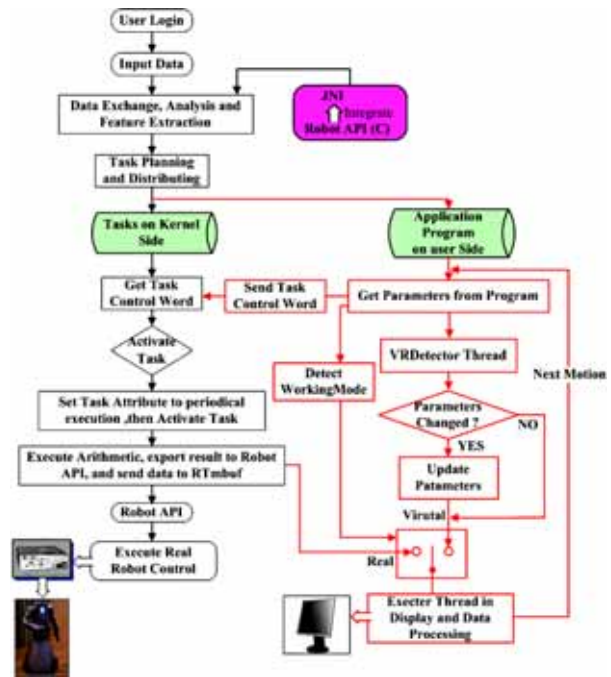


Figure 9 Real-Time Control Process in System

4 Conclusion

In this paper, a new system for Dual-Arm mobile robot simulation and control was presented. Developed in Java, it provides enterprise user low cost of cross-platform transplanting. Offline programming results can be previewed in simulation for future analysis. User uses the optimal results to control actuarial robot thus avoiding potential problems. Precise analysis and prediction will be possible basing on RT-Linux RealTime mechanism. It also serves as an easy and kind GUI for robot controller.

The presented system is easy to learn and use. Due to its easy programming panel, precise simulation and powerful control for actual robot, it seems to be very suitable for Dual-Arm mobile robot. It is highly desired in enterprises to create and maintain virtual installations of Dual-Arm robot applications throughout their life cycle.

Acknowledgement

We would like to express our gratitude to YASKAWA Electric Cooperation for offering the collaborative research funds.

References

- [1] Martin Rohrmeier, Web based robot simulation using VRML [C]//. Proceedings of the 2000 Winter Simulation Conference, vol.2., Dec. 2000: 1525-1528.
- [2] Melinda F Robinette, Rachid Manseur Robot-Draw. An Internet-Based Visualization Tool for Robotics Education [J]. IEEE Transactions on education (S0018-9359), Feb. 2001, 44(1): 29-34.

4 仿真结果

本文的仿真过程尽量满足过程的复杂性和模拟实际生产的各种可能发生的事件进行。如图 6 所示。

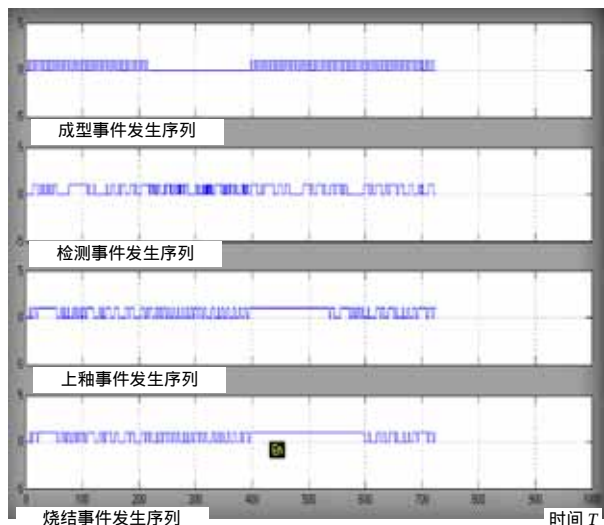


图 6 仿真结果

从上到下的波形分别是上层显示的运行结果: 成型状态、有无裂纹、上釉状态、烧结状态。我们可以根据仿真结果波形图来分析整个瓷砖制造的运行过程, 从而对它的状态进行实时监控, 得出各个加工程序的实时状况。下面分析图 6 的仿真结果。刚开始时(约 $t < 60$), 这是个启动整个制造的过程, 4 个图形输出波形的时刻有先后, 表明了这个工艺流程的发生顺序:

modeling_finish $\rightarrow$ destroyed $\rightarrow$ glazing_finish $\rightarrow$ sinter_finish, 约在 $60 < t < 220$ 后, 四个工艺都同时进行, 并没有故障发生, 并且我们可以看出图形发生周期性的变化, 表明这个加工过程在周而复始进行。当 $220 < t < 390$ 这段时间, 第一个图形 modeling_finish 无输出, 表明这个时候机器 1 发生了故障, 但是上釉和烧结仍在进行, 故障维修成功后, 图形开始有输出波形, 但是在 $395 < t < 440$ 这段时间, 第三个图形 glazing_finish 无输出, 表明机器 2 发生故障, 此时机器 3 处于待料状态或故障状态, 无输出。当机器 2 成功维修后, 恢复正常工作, 而图形三仍无输出, 显

(上接第 2545 页)

- [3] Vollmann K. A new approach to robot simulation tools with parametric components[C]// Industrial Technology, 2002. IEEE ICIT '02. 2002 IEEE International Conference, vol.2, 2002, 881-885.
- [4] Dixon WE, Moses D, Walker ID, Dawson DM. A Simulink-based robotic toolkit for simulation and control of the PUMA 560 robot manipulator[C]// Intelligent Robots and Systems, 2001, Proceedings, 2001 IEEE/RSJ International Conference, vol.4, 29 Oct. -3, Nov. 2001, 2202-2207.
- [5] Freund E, Roßmann J, Uthoff J, van der Valk U. Towards realistic

示了烧结过程发生了故障, 图形最后又是一个整个过程正常工作的仿真曲线。整个瓷砖制造的构建是从动态控制元结构开始, 通过串联组成车间级 Agent, 同时又包含了层次结构, 然后再与其他的动态控制元结构组成工厂级 Agent。

5 总结与展望

本文提出了动态控制元结构以及由其构建系统的理念, 根据离散事件动态系统的特点, 运用基于 SIMULINK/StatFlow 的仿真工具, 构建了整个系统并显示了整个系统的运行过程。仿真实验结果表明, 运用动态控制元结构的设计思想来构建系统, 是可以实现其能控、能观、无阻塞的受控性能的, 从而验证了我们的设计思想的正确性和可行性。

由于条件有限, 本论文只是用仿真来验证, 未能进行现场调试和检验, 可以在实际运行这个环节做研究; 另外, 可以建立能够表达状态、事件、转移函数、输入、输出以及区分离散事件与连续变量的编码规则, 从而建立可视的动态控制元结构类库; 在有限状态自动机研究的基础上, 可以以双向输出自动机构建 Agent 之间及系统不同层次之间的接口 Agent, 提出对系统进行静态结构特性和动态结构特性进行分析的完整算法。

参考文献:

- [1] 张小花, 陈玮, 邓则名. 基于 MAS 的分布式制造资源集成系统动态控制元结构的研究[J]. 组合机床与自动化加工技术, 2004, 12: 51-54.
- [2] 西北轻工业学院等编. 瓷砖工艺学[M]. 北京: 轻工业出版社, 1980.7.
- [3] 张葛祥, 李娜. Matlab 仿真技术与应用[M]. 北京: 清华大学出版社, 2003.
- [4] 薛定宇, 陈阳泉. 基于 Matlab/SIMULINK 的系统仿真技术与应用[M]. 北京: 清华大学出版社, 2003.
- [5] Ramadge P J.G, Wonham W.M. Supervisory control of a class of discrete event processes [J]. SIAM Journal on Control and optimization (S0363-0129), 1987, 25(1): 206-229.
- [6] Ramadge P J.G, Wonham W.M. The Control of Discrete Event Systems [J]. Proceedings of the IEEE(S0018-9219), 1989, 77(1): 81-98.
- [7] Simulation of Robotic Workcells [C]// Proceedings of the IEEE/RSJ/GI Intelligent Robots and Systems, Sep. 1994:39-46.
- [8] Sun Microsystems Inc. The Java 3DTM API Specification 1.3, [EB/OL]. (Jun. 2003). http://java.sun.com/products/java-media/3D/forDevelopers/J3D_1_3_API/j3dguide/index.html.
- [9] Victor Yodaiken, Michael Barabanov, RTLinux, Version Two, [EB/OL].(Nov.1999). <http://www.thinkingnerds.com/fsmllabs/archive/design.pdf>.
- [10] Denavit J, Hartenburg R S. A Kinematic Notation for Low-pair mechanisms Based on Matrices[J]. J. of Applied Mechanics (S0021-8944), 1955, 2: 215-2211.